Reputation Propagation and Updating in Mobile Ad Hoc Networks with Byzantine Failures

Chuanyou Li, Michel Hurfin, Yun Wang

Reputation Propagation and Updating in Mobile Ad Hoc Networks with Byzantine Failures

Chuanyou Li and Michel Hurfin
Inria, Rennes, France
Email: chuanyou.li@gmail.com
Email: michel.hurfin@inria.fr

Yun Wang
School of Computer Science and Engineering
Southeast University, Nanjing, China
Email: yunwang@seu.edu.cn

***Abstract*—In a mobile ad hoc network we consider the problem of designing a reputation system that allows to update and to propagate the computed reputation scores while tolerating Byzantine failures. Each time a correct node uses directly a service, it can determine by itself the quality of service currently provided. This fresh and valid rating information is broadcast immediately to all its current neighbors. Then, while the mobile node moves, it can receive from other nodes other recommendations also related to the same service. Thus it updates continuously its own opinion. Meanwhile it continues to broadcast this updated information. The freshness and the validity of the received/sent information become questionable. We propose a protocol that allows a node to ignore a second hand information when this information is not fresh or not valid. In particular, fake values provided by Byzantine nodes are eliminated when they are not consistent with those gathered from correct nodes. When the quality of service stabilizes, the correct nodes are supposed to provide quite similar recommendations. In this case, we demonstrate that the proposed protocol ensures convergence to a range of possible reputation scores if a necessary condition is satisfied by the mobile nodes. Simulations are conducted in random mobility scenarios. The results show that our algorithm has a better performance than typical methods proposed in previous works.**

***Keywords*-reputation; mobile ad-hoc network; byzantine;**

I. Introduction

In a mobile ad hoc network, nodes cooperate in a distributed and self-organized way to achieve some predefined goals. Thanks to a reputation mechanism, each node can benefit from the monitoring of the quality of service performed by others and can decide to interact or not with a service provider. A service provider is called a *trustee*, while a service user is called a *truster*. In this study we consider that a service is provided either by a fixed device located in a particular place or by a specific mobile node. In both cases, the user and the service provider must share the same location to be able to interact. We assume that a service user is necessarily a mobile node. Moreover, even if a service is not used by all the mobile nodes, the nodes which do not act as a truster/user (for this particular trustee/service) participate to the computation of its reputation score.

The quality of service provided by a trustee fluctuates all over the time. Of course, this instability can be the consequence of a failure affecting the trustee. But, even in the case of a correct trustee, the provided quality of service may evolve smoothly all the time (improvement or degradation phases). As a consequence, the reputation of a trustee computed by a mobile node is a belief level that has to be updated continuously. When a node uses a service, it can attest the quality of service provided by the corresponding trustee. This direct observation is performed by a user while it interacts with the trustee. At the end of the interaction, the rating provided by a correct truster is called a *first hand* information. This information is both fresh and valid. It can be shared immediately with the nodes located within the communication range of the user. Yet, broadcasting only first hand information is not sufficient: a user has a limited wireless communication range and cannot provide its rating to all the nodes. To propagate the information, each node (even those that are far from the trustee or those that have never used its service) must, on one side, broadcast its current estimate of the reputation of the trustee within its own neighborhood and, on the other side, update this estimation with the received values that seem to be fresh and valid. Thanks to this computation and propagation of *second hand* information, all the trusters can obtain indirectly a (more or less accurate) estimation of the recent quality of service provided by a given trustee.

We consider an unreliable ad hoc network. Some mobile nodes may suffer from Byzantine failures. By definition, these Byzantine nodes may behave arbitrarily. In particular, when they execute the reputation protocol, they may provide fake information (about the perceived quality of service or the freshness of their estimation). Identifying the received data that have to be ignored is a key problem. During its moving within the covered geographical area, a correct node gathers data provided by the (correct and faulty) encountered nodes. The proposed protocol allows to identify among these received estimations those that are not fresh enough or not valid. The computation of second hand information must rely only on messages that meet timeliness and validity requirements. To reach this goal, the protocol assumes an upper bound f on the number of Byzantine nodes. It postpones an update of the local reputation estimation until the node

has gathered a sufficient number of recent values (between $f+1$ and $2f+1$ values depending on the circumstances).

The main contributions of this paper are summarized as follows. First an algorithm to propagate and update the reputation scores is proposed. This protocol is designed to address three major issues namely: i) the participating nodes are mobile and the communication graph is constantly changing, ii) some nodes are Byzantine nodes and sometimes their inputs have to be ignored (validity requirement), and iii) out-dated values provided by correct or faulty nodes have also to be filtered out as the quality of service provided by a trustee is fluctuating (timeliness requirement). The proposed algorithm is inspired from a linear iterative consensus algorithm [1] which has been extended to fit the specific requirements of this different problem. The algorithm is distributed and all nodes perform asynchronous actions (*e.g.*, when a node updates its local reputation scores, other nodes are perhaps still moving to collect new messages). Second, we provide a theoretical analysis of the proposed protocol to demonstrate that some convergence properties can be satisfied if the dynamic topology satisfies a particular condition. Obviously, if a correct node remains isolated, it cannot update its estimations and thus convergence between all the nodes is impossible. The proposed condition aims to fix some additional constraints that have to be satisfied by the mobility scheme of all the correct nodes. Assuming that the quality of service provided by a trustee is stable and equal to b with $0 \leq b \leq 1$, when all the correct trusters (which may have subjective opinions) provide first hand information which are in the range $[a, c]$ with $0 \leq a \leq b \leq c \leq 1$, we demonstrate that the estimations of the reputation score established by the correct nodes converge also to the same interval if the identified condition is satisfied. Third, simulations are conducted. Based on two mobility model (random and regional random), we simulate our algorithm and also some other strategies that have been proposed in former works. The results show that the proposed algorithm reaches convergence and reach this objective more quickly.

The rest of this paper is organized as follows. Section II sketches out some related works. Section III introduces the system model. In Section IV, we establish a Bayesian Reputation System and then in Section V we give out the problem description. In Section VI, a new reputation dissemination and updating algorithm based on a linear iteration consensus method is proposed. In Section VII we focus on the reputation convergence and give some details about the condition that has to be satisfied. In Section VIII, we evaluate the performance through simulations.

II. Related works

Many works consider reputation systems in mobile ad hoc networks [2]: reputation propagation and updating are two main issues. The propagation allows to provide a recent knowledge to a node that has not yet (or not recently) interacted with a trustee. In [3], the propagation is based on flooding and the use of multiple routes. [4] proposes a method based on rendezvous. Yet, it requires that no consecutive Byzantine nodes are along a single communication path. This requirement is hard to ensure in a mobile network. In [5], trust is assumed to degrade automatically when the propagation hop length increases but in practice, such a degradation is not always observed. In all these works, only first hand information are propagated without modifying them while our solution relies on a mixing of the received values with the receiver's opinion.

In [6], all the nodes are correct and the network is assumed to be fault free. In the presence of malicious nodes, different strategies have been proposed to detect and suppress the invalid recommendations that can be generated during the propagation process. In [7], [8], a deviation test is implemented: a comparison is made between the value contained in a receive message and the corresponding local record. If the deviation is greater than a threshold, then the value is ignored. The solution proposed in [9] is based on the aggregation of multiple observations related to the same trustee: within the set of collected observations, abnormal gathered values are suppressed. Other works [10], [11] rely on the level of trust to identify the invalid recommendations. A received message is suppressed if the receiver considers that the sender is un-trusted (*e.g.* the corresponding local trust record is lower than a threshold). Yet, this solution is not robust enough. For example, a Byzantine node can provide a good quality of service to obtain a high level of trust, but when it has to relay recommendations, it may generate fake values that are not discarded.

Note that the two above methods do not consider a timeliness requirement. In a mobile environment, it could be the case that only a few trusters interact with a trustee during a period of time. In that case only these few nodes have fresh ratings. Yet, in the solutions described in [7], [8], this recent information can be ignored by other nodes if the deviation between the new value and the old ones exceeds the threshold. Similarly, in the solution described in [9], this new information provided only by a few nodes can also be deemed as abnormal and rejected. In [12] the authors point out that the reputation is timeliness. When new values are aggregated, time is considered as a forgetting factor.

III. Model

We consider a mobile ad-hoc network composed of n mobile nodes. An unique identifier p_i (with i ranging from 1 to n) is assigned to each node. The notation V represents the whole set of nodes: $V = \{p_1, p_2, ..., p_n\}$. Throughout the computation, the nodes move but still remain in a pre-defined limited geographical area. Collisions between the physical entities are not considered: several nodes can be located in the same place at the same time. No assumption is made about their trajectories: they can be random or predefined.

Some nodes may suffer from Byzantine failures. At any time a Byzantine node can behave arbitrarily: it may send fake messages, delay/drop messages or even stop its computation. Nevertheless, the adversary power of a Byzantine node is partially limited. We suppose that a Byzantine node can not use fake identities to communicate with other nodes. Therefore, a node can always identify the real sender of any message it receives. Thanks to this limitation, when duplicated messages originate from a same (correct or Byzantine) node, unnecessary messages can be easily detected and suppressed to keep only one of them (more precisely the one that seems to be the most recent one). Nodes that are not faulty are called correct nodes: they always follow the protocol's specification. The set of nodes V is partitioned into two disjoint subsets denoted V_c (the set of correct nodes) and V_b (the set of Byzantine nodes). We assume that an upper bound on the number of Byzantine nodes (denoted f) exists and is known. By definition, $\mid V_b \mid \leq f$ and $\mid V_c \mid \geq n - f$. Of course, the correct nodes don't known the number and the identities of the Byzantine nodes.

The wireless nodes communicate with each other only by exchanging messages. A node can only communicate with the nodes (called its neighbors) that are located within its limited radio communication range. Consequently, the communication graph is dynamic. As wireless communication can be interfered, some messages can be lost. Consequently, even if two correct nodes are neighbors, it could be the case that these nodes are not able to communicate or that only one of them successfully provide information to the other one. When the network is not heavy loaded, message losses are rare and a mutual exchange of information between neighbors is often possible. The message passing delay depends on many parameters (channel conditions, scheduling algorithm, arrival rate, etc). Works [13] on the propagation delay in ad hoc networks pointed out that the message transfer delay between two neighbors is usually in the order of a few milliseconds. In this study, when a message is successfully transferred, the delay can be ignored. Indeed all the upper bounds used in our protocol to identify time intervals are clearly of a different order of magnitude.

Nodes may start the computation at different times and no global scheduler synchronizes their actions. Yet, we require that the local clocks of the correct nodes are reasonably synchronized. The maximum difference between any two correct nodes' clock values must be very low compared to the upper bounds on time intervals used in the protocol.

Within the geographical area covered by the n nodes, some services are available and can be used by the mobile nodes. The set of m services is denoted $S = \{s_1, s_2, ..., s_m\}$. The rules for accessing a service have not to be known by the proposed protocol which just relies on the fact that a rating feedback is generated by the user after each interaction. This feedback is called a first hand information. The opinion expressed by a node p_i on the provided quality of service is based only on what p_i has observed directly.

Two main scenarios are possible. In the first case, each service s_x is available in a fixed location and can be used by a node p_i only when p_i is located in this specific place. In the second case, each service s_x is provided by a particular node p_y of V and can be used by a node p_i whenever their two paths intersect: p_i can access to the service s_x while it is in the communication range of p_y. Even if the proposed protocol covers the two cases, we focus mainly on the second scenario (especially during the simulation process). As a service s_x is provided by a node p_y, it could be the case that p_y is a Byzantine node. In that case the quality of service may be arbitrary. When p_y is correct, the provided quality of service is often quite stable and never random. Even when it is not stable, the deviations are not frequent and rather progressive. Usually, a correct node provides a degraded quality of service only when it faces resource scarcity (lack of memory storage, energy limitation, ...). Under such assumptions, the design of a reputation system (to predict the future quality of service based on some recent past observations) is a relevant approach.

To simplify the presentation of the protocol, we assume that a single service s_1 is available: m is equal to 1. As all the services are managed independently, there is no loss of generality. This unique service is offered by a node p_s. All the nodes of V are involved in the execution of the reputation protocol. Only a subset of the correct nodes is supposed to use the service and only some of them do this on a quite regular basis: those particular nodes belong to a subset of V_c denoted Θ.

IV. A Bayesian Reputation System

To compute the reputation score for the trustee p_s, a node p_i uses beta probability density functions. A beta distribution function is represented by a tuple parameter $\langle \alpha_i, \beta_i \rangle$ where α_i (respectively β_i) represents the amount of positive (respectively negative) ratings counted by p_i. The corresponding probability density function $f(x|\alpha_i, \beta_i)$ is expressed by the equation (1).

$$f(x|\alpha_i, \beta_i) = \frac{\Gamma(\alpha_i + \beta_i)}{\Gamma(\alpha_i)\Gamma(\beta_i)} x^{\alpha_i - 1}(1-x)^{\beta_i - 1} \qquad (1)$$

The probability variable x satisfies $0 \leq x \leq 1$ and $\alpha_i \geq 1$, $\beta_i \geq 1$. The probability expectation value of the beta distribution is given by:

$$E(x) = \frac{\alpha_i}{\alpha_i + \beta_i} \qquad (2)$$

R_{is} denotes the reputation data maintained by the node p_i. R_{is} corresponds to a tuple $\langle \alpha_i, \beta_i \rangle$. The corresponding reputation score is denoted $v(R_{is})$. This value is the probability expectation value computed with the formula (2). It is a real value which belongs to $[0, 1]$. The value $v(R_{is})$

represents the level of trust of p_i with regards to the service s_1 provided by p_s. If $v(R_{is})$ is close to 1, p_s is considered by p_i as being a good service provider. On the contrary, when $v(R_{is})$ is close to 0, p_i considers that p_s is a bad service provider. Initially, as p_i has no knowledge about the service s_1, R_{is} is set to $\langle 1, 1\rangle$. Thus $v(R_{is})$ is initially equal to 0.5.

During the computation, α_i and β_i will be updated when p_i obtains new informations. A new information can be generated by p_i itself when it uses directly the service. In that case, p_i which is close enough from p_s to be able to interact with it, generates a first hand information (denoted Fh_{is}) at the end of the interaction. A first hand informations reflects the opinion of the truster p_i about the quality of service provided by the trustee p_s. By assumption, this opinion which may be subjective is however rather close from the effective quality of service provided by p_s. The notation S_q represents the real quality of service provided by p_s when the service s_1 has been used for the last time by one node.

The reputation score does not evolve only when first hand information are locally generated. Each node also disseminates its current estimation of the reputation score. The information received by p_i from another node p_j is called a *second hand information* (denoted Sh_{js}). Of course, a Byzantine node can generate fake second hand information. Section VI details the protocol which updates the reputation score based on the first hand and second hand information.

V. Problem description

In a mobile ad hoc network, where up to f Byzantine nodes may exist, each correct node maintains an estimation of the reputation score which has to be as close as possible from S_q, the quality of service provided by p_s during its last interaction. By assumption, when a node p_i generates a first hand information, this objective is satisfied: at that time, Fh_{is} and S_q are values that are quite close. When the estimation becomes a mix of first hand and second hand informations, the following convergence property ensures that all the estimations computed by correct nodes converge.

Definition 1: (Convergence Property) When the quality of service provided by p_s is stable, all the new generated first hand informations belong to an interval $[MinFh, MaxFh]$ and eventually all the computed reputation scores belongs to the interval $[MinFh - \epsilon, MaxFh + \epsilon]$ (ϵ is a positive parameter close from zero).

VI. The Reputation Algorithm

Figure 1 depicts the variables managed by a correct node p_i involved in the reputation protocol. To simplify the notations, we no more refer to the identity of the unique trustee p_s (*e.g.* $FSh_{is} \longrightarrow FSh_i$, $Fh_{is} \longrightarrow Fh_i$, $Sh_{is} \longrightarrow Sh_i$). At a first layer, a correct node p_i uses a variable FSh_i to keep the value of either its last first hand information or a second hand information. FSh_i is a pair $[x, y]$ where x denotes the number of satisfactory interactions and y denotes the number of unsatisfactory interactions. At a second layer, p_i manages the reputation score R_{is}.

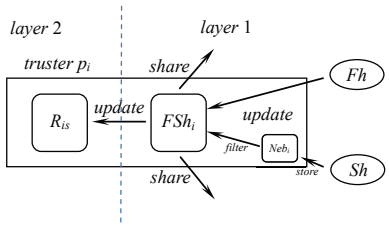


Figure 1. Variables managed by a node p_i at two different layers

Layer 1 is devoted to the operations on FSh_i. The value of FSh_i has to be propagated within the network. Therefore, p_i broadcasts periodically FSh_i to its current neigbhors to share its current knowledge. The value of FSh_i is updated by p_i in two different cases. First an update is done when p_i uses the service and generates a first hand information Fh_i. In that case, Fh_i becomes immediately the new value of FSh_i. When p_i receives enough second hand information, it can also update its variable FSh_i to mix its previous knowledge with the new received information that seem to be fresh and valid. Due to the fact that Byzantine nodes can propose unfair positive ratings or unfair negative ratings, this second type of update is more complex. Two main steps are required to identify fresh and valid second hand information. First out-dated information have to be suppressed. Then only the received information that are valid have to be selected. This selection process is inspired from a solution proposed to solve the linear approximate Byzantine consensus problem [1]. More precisely, all the received second hand informations are logged in a set denoted Neb_i where at most one message per external node is stored. When enough messages from different nodes are contained in Neb_i, the risky messages are identified and suppressed. To test the freshness of a value, the date at which the value has been computed by the sender is checked. Note that a Byzantine node can easily pass this first test by providing a recent date. But to circumvent the validity test, a Byzantine node has to provide a value that is rather similar to the fresh values provided by correct nodes. Indeed a second hand estimation v is considered as being a valid information if at least one correct node has proposed a fresh value lower than or equal to v and at least one correct node has proposed a fresh value greater than or equal to v. Finally all the remaining fresh and valid values are combined with the current value of FSh_i to compute the next value of FSh_i.

The pseudo code executed by a correct node p_i is provided in Figure 2. Layer 1 contains four tasks while layer 2 is composed of only one task. The lines 1-4 initialize the four main variables managed by truster p_i. Initially the reputation rating R_{is} is set to $\langle 1, 1\rangle$ to indicate that p_i feels uncertainty about the trustee s (line 1). FSh_i which is used

```
    % Initialization of node p_i
 1: R_is ← ⟨1, 1⟩;
 2: FSh_i ← ⟨1, 1⟩;
 3: t_i ← t_0;
 4: Neb_i ← ∅;
    % The protocol steps of Layer 1
    task 1:
 5: if ((clock_i() − t_0)%cycle equals to 0) then
 6:     p_i broadcasts ⟨FSh_i, t_i⟩;
 7: end if
    task 2:
 8: if (new Fh_i is obtained) then
 9:     FSh_i ← Fh_i
10:     t_i ← clock_i();
11:     Neb_i ← ∅;
12: end if
    task 3:
13: if (⟨FSh_j, t_j⟩ is received & t_j ≥ t_i ) then
14:     Neb_i ← ⟨FSh_j, t_j⟩ ∪ Neb_i;
15: end if
16: if (Neb_i is updated) then
17:     Neb_i ←sort(Neb_i);
18:     x ← the number of values bigger or equal than v(FSh_i);
19:     y ← the number of values less or equal than v(FSh_i);
20:     if (x ≥ f + 1 or y ≥ f + 1) then
21:         RNeb_i ← reducing(Neb_i, f);
22:         FSh_i ←update(FSh_i, RNeb_i);
23:         Neb_i ← ∅;
24:     end if
25: end if
    task 4:
26: Suppress the values in Neb_i received before clock_i() − Δ;
    % The protocol steps of Layer 2
27: if (FSh_i is updated) then
28:     R_is ← merge(R_is, FSh_i);
29: end if
```

Figure 2. Reputation updating algorithm

to keep either first hand or second hand information is also initialized to be $\langle 1, 1\rangle$ (line 2). Variables t_i (line 3) is a timestamp. Its initial value is t_0 which is assumed to be the starting time of the system. The line 4 initializes a buffer set Neb_i which is used to store received messages.

In Task 1 of layer 1 (lines 5-7), p_i periodically broadcasts FSh_i to its current neighbors. The function $clock_i()$ returns p_i's current local clock time. The variable $cycle$ identifies the predefined broadcasting period. When p_i sends a message with FSh_i, the timestamp t_i is also included to inform each receiver about the freshness of its estimation. Thus a receiver is able to test the freshness of the received estimations.

Task 2 (lines 8-12) is executed when a new first hand information is generated by p_i. During its interaction with p_s, we assume that p_i counts both the number of the satisfactory transaction and the number of the unsatisfactory transaction. These two informations are contained in Fh_i. For example, let us consider that message routing is the service provided by p_s. Suppose that p_i transmits 10 messages to p_s and then it observes that p_s forwards 8 messages among 10. In this particular case, p_i obtains 8 satisfactory transactions and 2 unsatisfactory transactions. Consequently, $Fh_i = [8, 2]$. When the service's interaction ends, p_i sets FSh_i to Fh_i (line 9) and updates t_i to the current local time (line 10). As p_i has the most up-to-date information, the second hand information contained in Neb_i become useless (line 11).

In task 3 (lines 13-25), when a new message $\langle FSh_j, t_j\rangle$ is received, p_i starts a selection process by comparing the timestamps (lines 13-14). If the timestamp t_j is equal to or greater than t_i, the new received message will be added into Neb_i, otherwise it will be discarded. As a Byzantine node can propose a fake value with a recent timestamp, for safety reasons, a message is not directly used to update FSh_i. In line 14, p_i stores the message into the set Neb_i. If a message from p_j already exists in Neb_i, the older one is suppressed. If the received message has been logged into Neb_i, the test performed at line 16 is satisfied. The end of task 3 (lines 17-24) allows to update FSh_i only when enough messages have been gathered. First (line 17), all the messages contained in Neb_i are sorted into an ascending order (according to the value of FSh_j contained in each message). In line 18 (respectively 19) the number of values which are greater (respectively smaller) than or equal to $v(FSh_i)$ is determined. The test evaluated by p_i at line 20 defines two favorable cases: either p_i has received values that are greater than or equal to FSh_i from at least $f + 1$ different nodes, or p_i has received values that are smaller than or equal to FSh_i from at least $f + 1$ different nodes. The result of the evaluation of this test depends mainly on the number of messages that have been received: (*a*). if p_i has received less that $f+1$ messages from different trusters, the test cannot be satisfied; (*b*) if p_i has received messages from at least $2f + 1$ different trusters, the test is necessarily satisfied; (*c*) if p_i has gathered more than f but less than $2f + 1$ messages, the test can be satisfied or not. When the test at line 20 fails, p_i does not update FSh_i now. It will continue to move and collect more messages. Otherwise, p_i starts the updating process. The procedure *reducing* (called at line 21) suppresses risky messages from Neb_i. If there are less than f messages with values greater than its own FSh_i, then p_i suppresses all these messages (greater than its own). Otherwise, p_i suppresses the largest f inside Neb_i. Likewise, if there are less than f messages with values smaller than its own FSh_i, then p_i suppresses all these messages (smaller than its own). Otherwise, p_i suppresses the smallest f inside Neb_i. After the reducing operation, the remaining values are copied into the set $RNeb_i$. At this stage, all the values are both fresh and valid. Either they have been proposed by a correct node or, when the sender is a Byzantine node, the proposed value is neither to small nor to large compared to all the fresh values provided by correct nodes. The node p_i updates FSh_i at line 22. Then p_i resets the buffer Neb_i to $\emptyset$ at line 23. The new value FSh'_i is the result of an arithmetic average.

$$FSh_i' = FSh_i + \sum_j a_j FSh_j \quad (3)$$

Initially, $FSh_i = [x_i, y_i]$. For each $FSh_j = [x_j, y_j]$ contained in $RNeb_i$, a parameter a_j equal to $(x_i+y_i)/(x_j+y_j)$ is used in the formula. As a consequence, if $|RNeb_i| = \kappa$, then $v(FSh_i') = (v(FSh_i) + \sum_j v(FSh_j))/(\kappa+1)$. Moreover, the following safety property is always satisfied.

Property 1: If $v(FSh_k)$ and $v(FSh_j)$ contained in Neb_i are respectively the minimum and the maximum value proposed by correct nodes, then after the execution of line 22: $\min\{v(FSh_k), v(FSh_i)\} \leq v(FSh_i') \leq \max\{v(FSh_j), v(FSh_i)\}$.

Task 4 (line 26) continuously eliminates old values stored in Neb_i. Parameter Δ is a given threshold which guarantees that no received message can be stored in Neb_i for more than the maximum duration time period Δ.

When FSh_i is updated, an update of R_{is} at layer 2 is also performed. The method $merge$ called at line 28 returns a new value $R_{is} = \lambda R_{is} + FSh_i$. The parameter λ is a forgetting factor and satisfies $0 \leq \lambda \leq 1$. When λ is close to 0, the new value of R_{is} is close to FSh_i. On the contrary when λ is close to 1, the new value is more strongly impacted by the past history.

VII. Convergence and Condition

When the quality of service S_q is stable, all the first hand informations $v(Fh)$ provided by the correct nodes are in a same interval $[MinFh, MaxFh]$. The size of this interval depends on the subjectivity of the opinions expressed by the correct nodes. The convergence property (definition 1) states that eventually all the computed reputation scores will also converge to this particular interval. In the algorithm, the reputation score is updated (line 28) when a new FSh is obtained locally. Even if the parameter λ slows down the convergence process, it is obvious that the convergence property is satisfied if eventually all the computed values $v(FSh)$ are in the expected interval. As the values $v(FSh)$ computed at line 9 are in this interval (by assumption), the proof has just to focus on the values $v(FSh)$ computed at line 22. At any given time t, among all the correct nodes, the minimum value and the maximum value of $v(FSh)$ are denoted $v^{min}[t]$ and $v^{max}[t]$ respectively. To prove the convergence property, we have to show that eventually $v^{min} > MinFh - \epsilon$ and $v^{max} < MaxFh + \epsilon$. In that case, convergence is reached. Analyzing the variations of v^{min} and v^{max} is complex because v^{min} (and in a similar way v^{max}) may fluctuate (increase and later decrease again due to the fact that older values are stored in logs Neb_i). To rely on a monotonic evolution of some values, we introduce the concept of "*window*".

Due to the reset executed at line 26, a value can stay in the log Neb_i during at most Δ units of time. Thus, at any time t_k, the oldest timestamp associated to a value contained in a log of a correct node is possibly less than t_k but greater than $forget(t_k)$. The function $forget(t_k)$ is defined as follows: if $t_k \geq \Delta$, then $forget(t_k) = t_k - \Delta$, otherwise $forget(t_k) = t_{-1}$ where t_{-1} is a virtual time corresponding to the time just before the execution starts. Based on the function $forget$, we introduce the concept of window. The window associated to time t_k is denoted $W[t_k]$. It corresponds to a non empty time interval $(forget(t_k), t_k]$. Suppose $W[t_k]/cycle = \eta$, thus at most $\eta(n-f)$ values owned by the correct nodes are proposed during $W[t_k]$. Among these $\eta(n-f)$ values we can identify the minimum and the maximum value. We denote them $v^{Wmin}[t_k]$ and $v^{Wmax}[t_k]$ respectively. Note that $W[t_k]$ contains time t_k, thus $v^{Wmin}[t_k] \leq v^{min}[t_k]$ and $v^{Wmax}[t_k] \geq v^{max}[t_k]$.

Lemma 1: At time t_k, if $MinFh > v^{Wmin}[t_k]$ then for any $t > t_k$, we have $v^{Wmin}[t_k] \leq v^{Wmin}[t]$. Similarly, if $MaxFh < v^{Wmax}[t_k]$, then for any $t > t_k$, we have $v^{Wmax}[t_k] \geq v^{Wmax}[t]$.

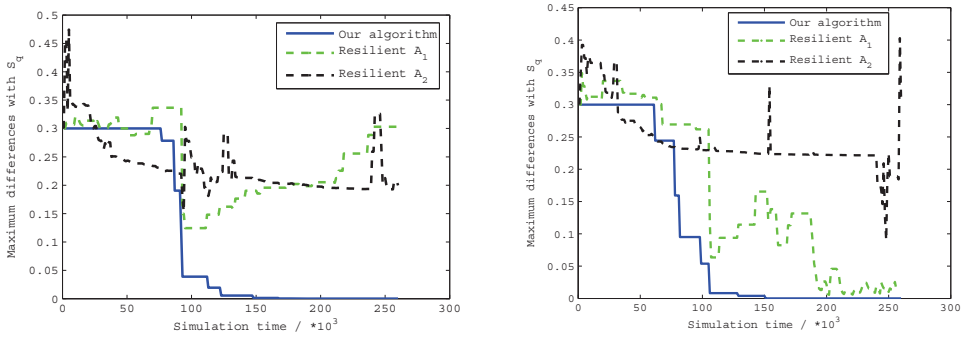
Lemma 2: At time t_k, if $MinFh \leq v^{Wmin}[t_k]$, then for any $t > t_k$, we have $MinFh \leq v^{Wmin}[t]$. Similarly, if $MaxFh \geq v^{Wmax}[t_k]$, then for any $t > t_k$, we have $MaxFh \geq v^{Wmax}[t]$.

Due to lemma 1 and lemma 2, when the quality of service is stable and all the first hand informations remain in the range $[MinFh, MaxFh]$, we conclude that $v^{Wmin} > MinFh - \epsilon$ and $v^{Wmax} < MaxFh + \epsilon$. The convergence property is satisfied.

To prove the convergence, a (necessary and sufficient) *condition* has also to be defined. The correct trusters act as sources of information and provides values that are in the range $[MinFh, MaxFh]$. Any correct node must receive (directly or indirectly) some recent first hand informations. Thus, while the convergence is not reached (either $v^{Wmin} < MinFh$ or $v^{Wmax} > MaxFh$), a condition has to be satisfied by the dynamic multi-hops network. No permanent partition should exist. The communication graph must ensure, for any cut of set V_c, that the information can flow from one part to another within a limited time period. Moreover, It is necessary to assume the existence of a core of at least $f+1$ correct nodes that use frequently the service. This subset of nodes is denoted Θ. During any period of time Δ, those particular nodes will meet the trustee p_s and interact with it at least once.

VIII. Simulation

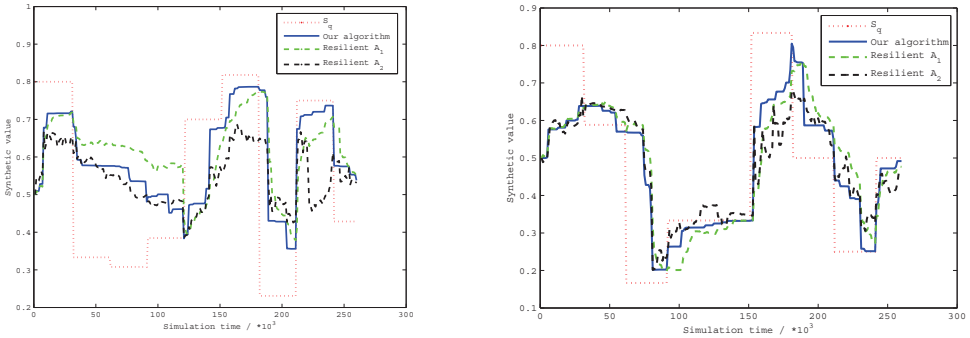
The simulation have been conducted on an open source platform *The one* [14], which is currently prevalent in DTN simulations. Each node plays simultaneously two roles "*trustee*" and "*truster*". Nodes move on a predefined graph $G = (V_g, E_g)$ represented by a square grid of size $l \times l$ ($|V_g| = l \times l$). We simulate two scenarios named respectively random move and regional random move. In both scenarios each node moves independently and can not move outside



(a) Random scenario

(b) Regional scenario

Figure 3. Simulation results - Convergence with a fixed value of S_q



(a) random / 30000 steps

(b) regional / 30000 steps

Figure 4. Simulation results - Convergence with a changing value of S_q

the graph. In random move, each node is initially located on a vertex randomly selected from G. When a node arrives at a vertex v_x, it starts to select randomly its next destination v_y such that $\langle v_x, v_y \rangle \in E_g$. The probability associated to the different choices are equivalent. In regional random move, nodes have different preferences. More precisely, G is divided into $Region_1$ and $Region_2$ ($Region_1 \cup Region_2 = G$). Correct nodes are also divided into two subsets V_c^1 and V_c^2. Initially, a correct node of V_c^1 (respectively V_c^2) is located on a vertex randomly selected from $Region_1$ (respectively $Region_2$). A node can select randomly its next destination, except when it is at the border between $Region_1$ and $Region_2$. In this case, a node of V_c^1 (respectively V_c^2) has a lower probability to move into $Region_2$ (respectively $Region_1$). The regional random model allows to test the case where some nodes have more chances to access a particular trustee than others. In both mobility scenarios, Byzantine nodes perform random move. Besides, we assume they have no stronger moving abilities (e.g. not a higher speed and no possibility to jump from one place to another).

Each node p_i provides a service with a quality of service $S_{q_i} \in [0, 1]$. It maintains two vectors $\vec{R_i}$ and $\overrightarrow{FSh_i}$. Inside the vector, each entrance (R_{ij} or FSh_{ij}, $j \neq i$) is used to keep information about the service provided by p_j. Initially, for all $j \neq i$, $R_{ij} = \langle 1, 1 \rangle$ and $FSh_{ij} = \langle 1, 1 \rangle$. p_i sends out $\overrightarrow{FSh_i}$ periodically. A Byzantine node may send out $\overrightarrow{FSh}$ with fake values. Nodes can receive $\overrightarrow{FSh}$ from another node only when they are within the same communication range (the losses of messages have not been simulated). When two correct nodes p_i and p_j exchanges messages (either p_i sends $\overrightarrow{FSh_i}$ to p_j, or p_j sends $\overrightarrow{FSh_j}$ to p_i), we assume the two nodes also interact and obtain a first hand information about each other. We assume that the rating $v(Fh)$ is always equal to the provided quality of service S_q. Each simulation lasts 259200 steps (simulate 72 hours).

Three different algorithms are tested and compared. The first one is our algorithm. The two others (named Resilient A_1 and A_2) are designed according to the related works presented in section II. When dealing with first hand information, A_1 and A_2 are similar to our algorithm. But the way they manage second hand information is different. A_1 is based on a deviation test. We set the deviation threshold to 0.1 which implies a correct node p_i will suppress a second hand information from p_j if $|v(FSh_i) - v(FSh_j)| > 0.1$. A_2 relies on trust levels to identify and merge compatible second

hand information. A correct node p_i suppresses the second hand informations that come from non-trusted nodes (if $v(R_{ij}) \leq 0.7$), otherwise it merges second hand information by weighted average: $FSh'_i = FSh_i + v(R_{ij})FSh_j$.

Analysis of the convergence when S_{q_1} is fixed (maximum differences: The grid size is fixed to 40×40 (1600 vertexes) and the length of each edge is set to 100 (thus, the size of the area is 4000×4000). 25 nodes (including 5 Byzantine nodes) are used in the simulation. Each of them has a different id (from 1 to 25). Nodes with the id from 21 to 25 are the Byzantine nodes. In regional random move, node 1 to node 7 prefer to move in $Region_1$ which occupies a quarter of G on the bottom side. Other correct nodes (id: 8 to 20) prefer to move in $Region_2$ which occupies three quarters of G on the top side. Every 10 steps, a node broadcast $\overrightarrow{FSh}$. The communication range is set to 50. All nodes have a same moving speed which is fixed to 30 per step. Figure 3 shows the results when node $id = 1$ is viewed as the trustee. As $v(Fh)$ is assume to be equal to S_{q_1}, we have $MaxFh = MinFh$ in this case. The figure shows the maximum differences between S_{q_1} which is fixed and $v(R_{i1})$ (with $p_i \in V_c$). Only our algorithm leads to convergence in both scenarios.

Analysis of the convergence when S_{q_1} evolves (aggregating rating): Reputation ratings from different correct nodes can be aggregated by simple vector addition of the components [9]. We still select node $id = 1$ as the trustee and simulate the variation of aggregating value. The settings of the simulation is almost the same as the simulation for the convergence, except that S_{q_1} is randomly changed during each 30000 simulation steps. At any time, $v(Fh)$ is assume to be equal to S_{q_1}. Thus the algorithm that provides a curve more close to the curve of S_{q_1} has a better performance. Figure 4 shows the results. Our algorithm (which identifies fresh and valid inputs) shows again a better performance. Furthermore, as the propagation mechanisms requires time to reach all correct nodes, our algorithm shows good performance when variation frequency of S_q decreases (the corresponding figures are not provided).

IX. Conclusion

We investigate the reputation propagation and updating problem in wireless mobile ad hoc networks with a few Byzantine failures. Based on linear consensus, an algorithm has been proposed which guarantees that only the messages with fresh and valid values are accepted. When first hand information is limited to the range $[MinFh, MaxFh]$, we investigate the convergence problem and provides a few hints about the sufficient and necessary condition. Simulation results show the interest of our algorithm.

Acknowledgment

This work is partially supported by 1) National 973 Hi-Tech Program, China, under Grant 61320605-3, and 2) by the ANR French national program for Security and Informatics (grant #ANR-11-INSE-010, project AMORES).

References

[1] Chuanyou Li, Michel Hurfin, and Yun Wang. Approximate byzantine consensus in sparse, mobile ad-hoc networks. *Journal of Parallel and Distributed Computing, Elsevier*, 74(9):2860–2871, 2014.

[2] Kannan Govindan and Prasant Mohapatra. Trust computations and trust dynamics in mobile adhoc networks: A survey. *IEEE Communications Survey & Tutorials*, 14(2):279–298, 2012.

[3] Yan Sun, Wei Yu, Zhu Han, and K. J. Ray Liu. Trust modeling and evaluation in ad hoc networks. In *Proc. of the IEEE Global Telecommunications Conference*, pp. 1862–1867, 2005.

[4] Ningning Cheng, Kannan Govindan, and Prasant Mohapatra. Rendezous based trust propagation to enhance distributed network security. In *Proc. of workshop on Security in Computers, Networking and Communications*, pp. 1066–1070, 2011.

[5] Sacha Trifunovic, Franck Legendre, and Carlos Anastasiades. Social trust in opportunistic networks. In *Proc. of IEEE Conf. on Computer Communications Workshops*, pp. 1–6, 2010.

[6] Yanbin Liu and Yang Richard Yang. Reputation propagation and agreement in mobile ad-hoc networks. In *Proc. of the IEEE Wireless Communications and Networking Conference*, pp. 1510–1515, 2003.

[7] Sonja Buchegger and Jean-Yves Le Boudec. A robust reputation system for mobile ad-hoc networks. *EPFL IC Technical Report IC/2003/50*.

[8] Charikleia Zouridaki, Brian L. Mark, and Marek Hejmo. Byzantine robust trust establishment for mobile ad hoc networks. *Telecommunication Systems*, 35(3-4):189–206, 2007.

[9] Andrew Whitby, Audun Josang, and Jadwiga Indulska. Filtering out unfair ratings in bayesian reputation systems. In *Proc. of the 3rd Int. Joint Conf. on Autonomous Agent and Multi Agent Systems*, pp. 106–117, 2004.

[10] Xueyuan Su, Gang Peng, and Sammy Chan. Forbid: Cope with byzantine behaviors in wireless multi-path routing and forwarding. In *Proc. of the IEEE Global Telecommunications Conference*, pp. 1–6, 2011.

[11] Guy Guemkam, Djamel Khadraoui, Benjamin Gateau, and Zahia Guessoum. Arman: Agent-based reputation for mobile ad hoc networks. In *Proc. of the 11th Conf. on Advances on Practical Applications of Agents and Multi-Agent Systems*, LNCS 7879, pp. 122–132, 2013.

[12] Jinshan Liu and Valerie Issarny. Enhanced reputation mechanism for mobile ad hoc networks. In *Proc. of the 2nd Conf. on Trust Management*, LNCS 2995, pp. 48–62, 2004.

[13] Shizhen Zhao, Luoyi Fu, Xinbing Wang, and Qian Zhang. Fundamental relationship between node density and delay in wireless ad hoc networks with unreliable links. In *Proc. Conf. on Mobile Computing and Networking*, pp. 337–348, 2011.

[14] Ari Keranen, Jorg Ott, and Teemu Karkkainen. The one simulator for DTN protocol evaluation. In *Proc. of the Conf. on Simulation Tools and Techniques*, 2009.